WHAT IS CLAIMED IS:

1. An interface between a microprocessor, or a local bus, and user macro-cells, being a macro-cell a self-consistent and pre-verified set of logic elements, generally designed with Hardware Description Languages, or similar, and transposed into silicon devices, the interface including a unique module connected between an external bus corresponding to the bus of said microprocessor, or being the local bus, and the remaining part of the interface, via an internal bus of the interface, said remaining part of the interface including peripheral resources, connected to the internal bus, named hereinafter common bus, and controlled by the unique module, named hereinafter main module, for the execution of read/write commands of the microprocessor towards a selected peripheral resource, wherein said peripheral resources are clustered in standardizable peripheral modules located externally to the interfaced macro-cells and connected to the macro-cells trough point-to-point buses, being each peripheral module in its turn constituted of a pre-defined set of hardware and firmware resources comprehensive of the most popular needs in interfacing user macro-cells, such as: configuration registers, command registers, status registers, not prefetchable registers for trapping events signalled by macro-cells, event counter registers, register synchronizers, dual port memories both of RAM and FIFO type either synchronous or asynchronous with respect to the clock that synchronizes the macro-cells.

2. Microprocessor, or local bus, interface in accordance with claim 1, wherein the pre-defined set of resources are partitioned inside each peripheral module into subsets of homogeneous resources, being each subset optionally equipped in accordance with specific needs of respective interconnected user macro-cells.

3. A microprocessor, or local bus, interface in accordance with claim 2, wherein said subsets of pre-defined set of resources include means for calculating a remote filling status of the associated resources expressed as a number of read/write individual transactions before a resource either becomes empty in a read transaction or full in a write transaction, said means for calculating the remote filling status being arranged to limit the calculated value at an integer greater than or preferably equal to the maximum round-trip delay of the transactions through the common bus expressed in number of master clock cycles.

4. A microprocessor, or local bus, interface in accordance with claim 3, wherein said main module includes means for currently tracing the remote filling status received from a selected peripheral resource in order to compensate locally to the main module the subsequent latency other than the initial delay, and signalling on the external bus either the

existence of room for transferring data or the opposite condition to terminate a current transaction.

5. A microprocessor, or local bus, interface in accordance with claim 4, wherein said means for currently tracing the remote filling status received from a selected resource includes in its turn:

- means for memorizing the received filling status for two consecutive master clock cycles;
- means for calculating a delta filling status between the received filling status at the current master clock cycle and the previously memorized filling status value, in so detecting variation of residual room either for write or read a datum in/from a not prefetchable peripheral resource; and
- means for calculating a local filling status by summing up the received filling status at the actual master clock cycle with the delta filling status and further subtracting a unity value in case a datum has been transferred from the external bus to the main module during a write or when the acknowledge to a read operation has been received from the COMMON-BUS;
- arithmetic means for saturating the local filling status at the number that can be represented with that bus width;
- means for detecting either a positive or null value of the local filling status as conditions for respectively enabling or terminating a current data transfer to/from the external bus.

6. A microprocessor, or local bus, interface in accordance with claim 1, wherein said main module includes means for generating command on the common bus arranged in a way to generate a command consisting of a pure query issued to an selected peripheral resource to return its remote filling status value, or a read-and-query, or a write-and-query, whose argument is a datum coupled with said remote filling status.

7. A microprocessor, or local bus, interface in accordance with claim 1, wherein said main module further includes:

- a centralized FIFO memory acting as a buffer shared among the peripheral modules for transitorily storing a burst of data read in advance from a peripheral selected subset of resources in correspondence of a command asserted to transfer data on the external bus;
- centralized FIFO memory control means in its turn including: a) means for calculating the filling status of the centralized FIFO memory in order to verify room to accept new data; b)

first counter and comparator means for counting the number of data words transferred from a selected subset of peripheral resources to the centralized FIFO memory and comparing a first counting value issued from said first counter means with a first prefixed threshold placed near the filling boundary of the centralized FIFO memory in order to avoid buffer overflow; c) means for purging the unread data from the centralized FIFO memory at the end of a current transaction towards the external bus in order to assign an empty buffer at the successive requester;

- second counter and comparator means for counting the number of data words transferred from the centralized FIFO memory to the external bus and compare a second counting value issued from said second counter means with a parameter indicating the length of a read data burst, in order to detect equality as a condition to terminate a current read burst transaction from the centralized FIFO memory to the external bus.

8. A microprocessor, or local bus, interface in accordance with claim 1, wherein said main module further includes:

- first interfacing circuit means connected to the external bus for receiving, transmitting, buffering, sampling, or holding address, data and control signals from/to the external bus;
- second interfacing circuit means connected to the common bus for receiving, transmitting, buffering, sampling, or holding address, data, and control signals from/to the common bus;
- address translator means cascaded with centralized address decoder means the cascade being connected between said first and second interfacing circuit means to select a peripheral resource for enabling a communication path between the microprocessor, or local bus, and an interconnected macro-cell;
- address space configuration means, active at the boot time after power up, to supply a granular memory map of the interface to said centralized address decoder means which is enabled consequently to translate the linear address space of the external bus into a structured address space of the common bus according to the physical topology and implemented by forwarding to the cluster of peripheral modules a full-range address bus accompanied with a lower-range sub-bus for the selection of said peripheral modules and the type of embedded resources through a two-steps address decoding, being the first step charged to said centralized address decoder means;

- internal arbitration means for receiving transaction requests from peripheral modules connected to transaction requesters internal to the respective user macro-cells, and generating a grant signal towards a selected requester to command the requester to transfer parameters suitable to sustain DMA burst transaction;
- plug-in means for de-coupling the protocol active on the external bus from the protocol active on the common bus;
- two-way transaction control means connected to the preceding means and promoting a communication protocol active on the common bus for allowing ordinate transactions through the two directions of the interface.

9. Microprocessor, or local bus, interface in accordance with the claim 7, wherein each standardizable peripheral module, further includes a peripheral controller including in its turn:

- first peripheral selector means for transferring towards the common bus either read data and the associated remote filling status, or the filling status only, returned from an addressed resource;
- distributed address decoding means receiving said full-range and lower-range address buses of the common bus to complete the second address decoding step for the selection a memory or FIFO port or a register port inside the subset;
- protocol command decoding and sequencing means connected to the common bus to decode and properly time and forwards commands issued by the main module, and generate for each decoded command a respective peripheral monitoring code suitable to inform the main module about the operational state of the addressed resources;
- protocol control means cascaded to the preceding sequencing means for generating control and selection signals selectively forwarded to the subset of resources to address them for executing the decoded commands on the addressed resources;
- second peripheral selector means to forward towards the common bus the transaction requests generated from the macro-cells and selected parameters required by a specific grant cycle involving burst transactions;
- means for introducing a prefixed delay between its input signals constituted by the read data, the filling status and the burst transaction parameters, before transferring them on the common bus, being the delay equal to their known initial latency, in order to synchronize them with the peripheral monitoring codes.

10. A microprocessor interface in accordance with claim 9, wherein said protocol command decoding and sequencing means is further equipped with a peripheral partial flow-control-logic for receiving the remote filling status selected by said first peripheral selector means and decode it to return peripheral monitoring code to the main module either:

- a datum coupled with the value of the remote filling status, the peripheral monitoring code indicating a data-and-query payload, in correspondence of both a read-and-query decoded command and a remote filling status greater than zero; or
- the only remote filling status null in correspondence of both a read-and-query decoded command and a remote filling status null coupled with the peripheral monitoring code indicating a pure query payload.

11. A microprocessor, or local bus, interface in accordance with claim 2, wherein a first subset of said pre-defined set of resources includes a bank of memory registers comprising: configuration registers, command registers, event counter registers, status registers, read-and-reset registers.

12. A microprocessor, or local bus, interface in accordance with claim 11, wherein said first subset of resources of register type further includes multiplexing and demultiplexing means for selecting a specific register inside the bank of registers and transmitting towards said peripheral controller the register bank's filling status concerning read/write transactions, taken alone, or either accompanied with read or write data word from/to the selected register; being a served user macro-cell connected to one or more registers of the bank through point-to-point buses for writing or reading the various registers autonomously from the peripheral controller.

13. A microprocessor interface in accordance with claim 12, wherein a not prefetchable read-and-reset register is written from the macro-cells and successively read and reset from the main module in two separate steps that turn the register into a prefetchable one, being a first step devoted to read the register content and transfer it inside the centralized FIFO memory without altering the content into the read-and-reset register, and a second step for clearing up the content previously written into the register upon a condition that the content has been definitely transferred to the external bus agent.

14. A microprocessor, or local bus, interface in accordance with claim 2, wherein a second subset of said pre-defined set of resources includes prefetchable memory resources.

15. A microprocessor, or local bus, interface in accordance with claim 14, wherein said

second subset of resources of prefetchable memory type includes a dual port memory RAM having:

- a first read/write port connected to a read/write port of the peripheral controller for transmitting the filling status concerning read/write transactions and transferring data on the two directions of the common bus, and
- a second read/write port connected to a user macro-cell through an point-to-point bus for independently transfer data from/to the connected macro-cell.

16. A microprocessor, or local bus, interface in accordance with claim 2, wherein a third subset of said pre-defined set of resources includes not prefetchable memory resources.

17. A microprocessor, or local bus, interface in accordance with claim 16, wherein said not prefetchable resource of memory type includes:

- a first peripheral FIFO memory having a read port connected to a read port of said peripheral controller for transmitting the filling status and transferring read data on the common bus, and a write port for receiving write data originated from the user macro-cell through an interconnected point-to-point bus;
- a second peripheral FIFO memory having a write port connected to a write port of said peripheral controller for transmitting the filling status and receiving write data from the common bus, and a read port for transfer read data to the user macro-cell through an interconnected point-to-point bus.

18. A microprocessor, or local bus, interface in accordance with claim 2, wherein a third subset of said pre-defined set of resources includes:

- a modified FIFO peripheral resource, either synchronous or asynchronous with respect to the clock that synchronizes the interconnected macro-cell, coupled to a logic for preventing the read data be overwritten inside the modified FIFO until they are definitely transferred to the external bus;
- a not prefetchable memory resource.

19. A microprocessor, or local bus, interface in accordance with claim 18, wherein:

- said modified FIFO peripheral resource includes a read port connected to a read port of said peripheral controller for transmitting the filling status and read data on the common bus, and a write port for receiving write data originated from the user macro-cell through

an interconnected point-to-point bus;

- said not prefetchable resource of memory type includes a known peripheral FIFO memory having a write port connected to a write port of said peripheral controller for transmitting filling status and receiving write data from the common bus, and a read port for transfer read data to the user macro-cell through an interconnected point-to-point bus.

20. A microprocessor, or local bus, interface in accordance with claim 19, wherein said modified FIFO peripheral resource includes a synchronous dual port RAM for writing and reading data under control of an interconnected prefetchable FIFO controller capable to manage the dual port RAM as a circular buffer of FIFO type keeping stored in the RAM and conditionally not overwriteable said data popped out from the modified FIFO read port to be copied into said centralized FIFO memory, until the reception at the modified FIFO read port either a flush or clear-and-flush command, both accompanied with an associated value indicating the number of words to be flushed out because definitely transferred to the external bus, and the clear-and-flush command additionally indicating the end of the current transaction, in a way that the tandem of the modified FIFO and the centralized FIFO is suitable to allow anticipative reads for reducing subsequent latency in burst transactions without losing read data when the centralized buffer input is switched to service another peripheral subset.

21. A microprocessor interface in accordance with claim 20, wherein said prefetchable FIFO controller further includes a pointer-generating and handling logic prompted by said flush or clear-and-flush command and by additional pushword and popword commands to respectively write data into the modified FIFO write port and read data from the read port and consequently setting up:

- an aligned read pointer for tracing a readable location in the circular buffer adjacent to the location of the last datum overwriteable by effect of said flush command, in such a way that the possible data overwriteable are the only data definitely transferred to the external bus;

- a misaligned read pointer for tracing a readable location in the circular buffer adjacent to the location of the last datum copied into the centralized buffer by effect of multiple popword commands , in such a way that in the same transaction no data are read more than once; being the misaligned read pointer aligned to the aligned read pointer location at the reception of the clear-and-flush command;

- a write pointer for tracing the next writeable location into said circular buffer.

22. A microprocessor interface in accordance with claim 21, wherein said pointer-generating and handling logic includes the following means for generating a rewind command active on the aligned read pointer in concomitance with the execution of each flush command of the current transaction:

- a popword counter incremented by the popword commands and reset by the flush command;
- first subtracting means for subtracting the number of flushed out words from the value reached by the popword counting, obtaining a number of rewindable memory locations into the circular buffer whose content is prevented to be overwritten;
- second subtracting means for subtracting the number of rewindable memory locations from the current value of the aligned read pointer, in a way to update the offset between aligned and misaligned read pointers.

23. A microprocessor interface in accordance with claim 22, wherein said pointer-generating and handling logic further includes:

- a first presettable up/down counter for tracing the number of readable locations into said circular buffer, coupled to a supervisor control logic for generating an aligned counting value which counts data available for reading in the circular buffer considering readable data unread from the centralized since last flush command and the rewinded data to the last flush, the aligned counting value being suitable for generating said aligned read pointer;
- a second presettable up/down counter for tracing the number of readable locations into said circular buffer, coupled to the supervisor control logic for generating a misaligned counting value which counts data available for reading in the circular buffer considering readable only data not yet read from the centralized buffer, the misaligned counting value being suitable for generating said misaligned read pointer;
- a third presettable up/down counter for tracing the number of writeable locations into said circular buffer, coupled to the supervisor control logic for generating a writing counting value suitable for generating said write pointer.

24. A microprocessor interface in accordance with claim 23, wherein said prefetchable FIFO controller further includes circular buffer interfacing means connected between said

pointer-generating and handling logic and said dual port RAM for receiving both said misaligned read pointer and the write pointer and generate correspondent writing and reading addresses for the dual port RAM, and further to transmit write data and receive read data to/from the dual port RAM.

25. A microprocessor, or local bus, interface in accordance with claim 8, wherein said internal arbitration means is arranged in a way to generate an authorisation signal to enable said plug-in means to compete before an external arbiter for gaining the main module master-ship of the external bus, upon condition that at least one said transaction requests is detected at the input during the current master clock cycle, and for enabling the issuing of said grant signal to a selected requester on the basis of the following two alternative embodiments:

- at the only reception of said transaction requests; or
- upon detection of a back confirmation signal stating that the main module have gained the master-ship of the external bus.

26. A microprocessor, or local bus, interface in accordance with claim 25, wherein said internal arbitration means includes:

- a time slice counter for measuring the duration of a granted service period;
- a circular queue to push sequentially all service requests coming from user macro-cells;
- a scheduler to pop out one first request for service in conformity to a policy First-Come-First-Served, being the popped request signaled through the assertion of a correspondent grant signal ;
- a priority ROM to store at the i-nth location a number of clock pulses that a peripheral resource i-nth shall keep the grant asserted;
- a data comparator to compare, at every clock time, the counting of time slice counter with the contents of the i_nth row of the priority table corresponding to the actual granted request i, in order to stop the current data transfer in case of coincidence, and command the scheduler to pop out the next request for service at the i+1_nth row.

27. A microprocessor, or local bus, interface in accordance with claim 8, wherein said plug-in means includes the following means:

- a command decoder for decoding commands received from the external bus and forward

it to said two-way transaction control means;

- an address/command generator for generating address and commands to be issued on the external bus for read/write data at external destination addresses;
- an external bus sequencer connected to both the command decoder and the address/command generator suitable to implement the communication protocol active on the external bus and communicate with an external arbiter for requesting the main module master-ship of the external bus.

28. A microprocessor, or local bus, interface in accordance with claim 7, wherein said two-way transaction control means belonging to the main module includes centralized logic means for synchronizing the transactions between the external bus and the common bus by exploiting sub-means suitable for the following operations:

- detecting start/end transactions on the two buses,
- detecting the insertion of wait states on the two buses,
- detecting availability of data on the two buses,
- counting the data words transferred to the external bus,
- sustaining a direct memory access originated by an transaction requester internal to a respective user macro-cell to transfer data from the macro-cell towards an device connected to the external bus, being the transaction requester connected to the remote register subset for transferring parameters suitable to sustain the DMA transaction;
- storing said parameters suitable to sustain DMA burst transactions;
- monitoring the number of data currently stored in the centralized FIFO memory;
- monitoring the operations requested by an external bus protocol agent and promoting corresponding operations inside said main module of the interface;
- receiving and interpreting peripheral monitoring codes transmitted from the remote peripheral resources;
- receiving and interpreting a set of monitoring signals concerned transactions between the main module of the interface and the external bus.

29. A microprocessor, or local bus, interface in accordance with claim 28, wherein said two-way transaction control means belonging to the main module further includes the

following control means for co-operating with said centralized logic means:

- a main common bus sequencer for generating command on the common bus;
- a main handshake sequencer for generating status signals and commands towards said plug-in means, directed to both the embedded address/command generator and the external bus sequencer;
- a main module controller for generating control signals for all the means embedded in the main module of the interface other than the two-way transaction control means.

30. A microprocessor, or local bus, interface in accordance with claim 28, wherein the output of said centralized FIFO memory is further connected to:

- said first interface circuit means to pop out a datum indicating the source starting address of the reading burst;
- said address/command generator to pop out a datum indicating the destination starting address of the reading burst;
- a direct memory access controller embedded in said centralized logic means to pop out a datum indicating the length of a read burst executed exploiting the centralized FIFO memory.

31. A microprocessor interface in accordance with claim 11, wherein when the clock of an interfaced macro-cell, named hereinafter appl_clk, is asynchronous with respect to the common bus clock, named hereinafter cbus_clk, the latter timing the subset of register resources, then suitable synchronization circuits are embedded in corresponding peripheral modules and cascaded to as many registers connected to the asynchronous macro-cells, in order to ensure reliable communication between the two asynchronous clock domains at the two sides of the synchronization circuits.

32. A microprocessor interface in accordance with claim 31, wherein a synchronized status register between two asynchronous clock domains includes:

- first select and hold means for the selection of either a status word incoming from a macro-cell belonging to the appl_clk clock domain, or a previous stored status word outputted from the same means in the same clock domain;
- first flip-flop means of set-reset type for the generation of a control signal of said first select and hold means both enabling the selection of one said incoming status word in

presence of a write strobe accompanying the status word and blocking the selection of new status words until the reception of a release command;

- a first strobe synchronization circuit able to transfer a signal between two asynchronous clock domains for transferring said write strobe towards the cbus_clk clock domain;
- second select and hold means controlled by the write strobe transferred into the cbus_clk clock domain for selecting, storing, and forwarding towards the common bus either a status word outputted from said first select and hold means or the previous content of the same second select and hold means;
- a release circuit for generating the release command for removing the blocked condition settled by said first flip-flop means allowing the acquisition of a new status word incoming from the macro-cell.

33. A microprocessor interface in accordance with claim 32, wherein said release circuit includes:

- first flip-flop means of D type for sampling the write strobe transferred into the cbus_clk clock domain;
- a second strobe synchronization circuit identical to the first one to transfer the sampled write strobe towards the appl_clk clock domain and obtain said release command.

34. A microprocessor interface in accordance with claim 32, wherein said release circuit includes a second strobe synchronization circuit identical to the first one to transfer a read strobe incoming from the common bus towards the appl_clk clock domain, obtaining said release command.

35. A microprocessor interface in accordance with claim 32, wherein said release circuit includes:

- first flip-flop means of D type for sampling the write strobe transferred into the cbus_clk clock domain;
- a two-inputs select means for selecting either the sampled write strobe or a read strobe incoming from the common bus, obtaining a selected write strobe;
- a second strobe synchronization circuit identical to the first one to transfer the selected write strobe towards the appl_clk clock domain, obtaining said release command.

36. A microprocessor interface in accordance with claim 32, wherein said synchronized

status register further includes a two-inputs AND type gate receiving at one input a signal indicating said blocked condition settled by said first set-reset flip-flop means and at the other input a bit mode for either send back the signal indicating the blocked condition towards macro-cell, or not.

37. An interface protocol to manage two-way transactions between a microprocessor bus, or a local bus, and user macro-cells via an interposed interface; the protocol including a plurality of concurrent steps synchronized at the master clock cycle of the interface for carrying out read/write transactions to transfer data between the two sides of the interface through a main module of the interface connected both to an external bus coinciding with the microprocessor bus, or being the local bus, and an internal bus of the interface, named hereinafter common bus, further connected with prefetchable and not prefetchable peripheral resources constituting the remaining part of the interface coupled to the macro-cells; the main module having indifferently the mastership of the external bus, or being slave, in both cases issuing protocol commands on the common bus and receiving back correlated replies, further exchanging arbitrate signals with an external arbiter and handshake signals with an external bus protocol agent, wherein a write transaction to transfer one or more data from the external bus to a peripheral resource assigned to a subset of homogeneous resources belonging to one out of a plurality of standardizable peripheral modules clustered on the common bus, includes at least the following concurrent steps charged to the main module, each spanning preferably a period of said master clock cycle:

a) issuing on the common bus a wait-write-and-query command at the start of a new write transaction towards an addressed peripheral resource to have returned on the common bus a remote filling status stating the number of memory locations before the addressed peripheral resource become full; and looping the following steps until detecting the assertion of a terminating signal to end the write transaction:

b) calculating locally to the main module a local filling status by subtracting a unitary value to said returned remote filling status in case a datum has been transferred in the current step from the external bus to the main module;

c) checking whether said local filling status is greater than zero and either assuming a positive value as a condition for asserting an handshake ready signal causing a datum be transferred from the external to the common bus, or assuming a null local filling status value as a condition for issuing a command to terminate the current write transaction;

d) checking whether said local filling status is greater than zero and either assuming a positive value as a condition for issuing to the addressed resource a write-and-query command to transfer a datum from the common bus to a peripheral resource coupled to the destination macro-cell and to have returned on the common bus the remote filling status of the addressed resource, or assuming a null local filling status value as a condition for issuing a wait-write-and-query command for inserting a wait cycle on the common bus and have returned the remote filling status.

38. An interface protocol to manage two-way transactions between a microprocessor bus, or a local bus, and user macro-cells via an interposed interface; the protocol including a plurality of concurrent steps synchronized at the master clock cycle of the interface for carrying out read/write transactions to transfer data between the two sides of the interface, through a main module of the interface connected both to an external bus coinciding with the microprocessor bus, or being the local bus, and an internal bus of the interface, named hereinafter common bus, further connected with prefetchable and not prefetchable peripheral resources constituting the remaining part of the interface coupled to the macro-cells; the main module having indifferently the mastership of the external bus, or being slave, in both cases issuing protocol commands on the common bus and receiving back correlated replies, further exchanging arbitrate signals with an external arbiter and handshake signals with an external bus protocol agent, wherein a read transaction to transfer one or more data from a prefetchable peripheral resource to the external bus, being the prefetchable resources assigned to a subset of homogeneous resources belonging to one out of a plurality of standardizable peripheral modules clustered on the common bus, includes at least the following concurrent steps charged to the main module each spanning preferably a period of said master clock cycle:

a) issuing on the common bus a wait-read-and-query command at the start of a new read transaction towards an addressed peripheral resource to have returned on the common bus a remote filling status stating the number of memory locations before the addressed peripheral resource become empty; and looping the following steps until detecting the assertion of a terminating signal to terminate the read transaction:

b) calculating locally to the main module a local filling status by subtracting a unitary value to said returned remote filling status in case a datum has been transferred in the current step from the main module to the external bus;

c) checking if said local filling status is greater than zero; checking if a further filling status

stating the number of memory locations before a centralized memory FIFO inside the main module become full to write or empty to read is lower than a precautionary filling-up threshold; taking both the checked conditions true as a criterion for issuing to the addressed peripheral resource coupled to the data originating macro-cell a read-and-query command to have returned on the common bus a single datum coupled to the remote filling status of the addressed peripheral resource, being the returned datum transferred directly to the centralized memory FIFO; whether the criterion is not true issuing a wait-read-and-query command for inserting a wait cycle on the common bus and have returned the remote filling status;

d) checking if at least a valid datum to be read is present inside said centralized memory FIFO and if that is true, asserting an handshake ready signal for transferring the datum to be read from said centralized memory FIFO to the external bus;

e) checking if said local filling status and the reading filling status of the centralized memory FIFO are both null and, if that is true, asserting a signal to terminate the current read transaction;

f) checking if said terminating signal is asserted and, if that is true, issuing a command for clearing out the centralized memory FIFO content at the end of the current read transaction, in such that an empty centralized buffer is assigned at a successive peripheral requester.

39. An interface protocol to manage two-way transactions between a microprocessor bus, or a local bus, and user macro-cells via an interposed interface; the protocol including a plurality of concurrent steps synchronized at the master clock cycle of the interface for carrying out read/write transactions to transfer data between the two sides of the interface, through a main module of the interface connected both to an external bus coinciding with the microprocessor bus, or being the local bus, and an internal bus of the interface, named hereinafter common bus, further connected with prefetchable and not prefetchable peripheral resources constituting the remaining part of the interface coupled to the macro-cells; the main module having indifferently the mastership of the external bus, or being slave, in both cases issuing protocol commands on the common bus and receiving back correlated replies, further exchanging arbitrate signals with an external arbiter and handshake signals with an external bus protocol agent, wherein a read transaction to transfer one or more data from a prefetchable peripheral resource to the external bus, being the prefetchable resources assigned to a subset of homogeneous resources belonging to one out of a plurality of

standardizable peripheral modules clustered on the common bus, includes at least the following concurrent steps charged to the main module each spanning preferably a period of said master clock cycle:

a) issuing on the common bus a read-and-query command at the start of a new read transaction towards an addressed peripheral resource to have returned on the common bus either a datum coupled with a remote filling status stating the number of memory locations before the addressed peripheral resource become empty, or the only filling status null; and looping the following steps until detecting the assertion of a terminating signal to terminate the read transaction:

b) calculating locally to the main module a local filling status by subtracting a unitary value to said returned remote filling status in case a datum has been transferred in the current step from the main module to the external bus, by means of an arithmetic which bound each new calculated value of the local filling status to be greater or equal to zero;

c) checking if a further filling status stating the number of memory locations before a centralized memory FIFO inside the main module become full to write or empty to read is lower than a precautionary filling-up threshold, taking the true logic condition as a criterion for issuing to the addressed peripheral resource coupled to the data originating macro-cell a read-and-query command to have returned on the common bus either a datum coupled with a remote filling status stating the number of memory locations before the addressed peripheral resource become empty, or a remote filling status null, being the returned datum transferred to the centralized memory FIFO; whether the criterion is not true issuing a wait-read-and-query command for inserting a wait cycle on the common bus and have returned the remote filling status;

d) checking if at least a valid datum to be read is present inside said centralized memory FIFO and if that is true, asserting an handshake ready signal for transferring the datum to be read from said centralized memory FIFO to the external bus;

e) checking if said local filling status and the reading filling status of the centralized memory FIFO are both null and, if that is true, asserting a signal to terminate the current read transaction;

f) checking if said terminating signal is asserted and, if that is true, issuing a command for clearing out the centralized memory FIFO content at the end of the current read transaction, in such that an empty centralized buffer is assigned at a successive

peripheral requester.

40. An interface protocol in accordance with claim 38, wherein inside the reading loop:

- a suitable residual read burst length counting is provided to further support read data burst transactions originated from a not prefetchable peripheral resource of FIFO type receiving data from a FIFO embedded into a macro-cell, the reading burst having a known length;
- another concurrent step is provided in which the criterion stated at step c) is put in logical AND with a further condition stating that residual read burst length is greater than zero, being the new logical AND true as a global condition for issuing the read-and-query command;
- the residual read burst length counting being unitary decremented for each new datum has been read.

41. An interface protocol in accordance with claim 38, wherein suitable flushing commands are provided to further support read data burst transactions originated from a traditional FIFO embedded into a macro-cell upstream a modified FIFO peripheral resource coupled to a logic which prevents data read in advance from the modified FIFO be overwritten until the reception of said flushing commands, and inside the reading loop:

- an additional concurrent step similar to the step c) is provided and the new step differing from step c) mainly because said criterion is put in logical AND with a further condition stating that a prefixed number of data words have been definitely transferred to the external bus, being the true value of the new logical AND a global condition for issuing instead of the read-and-query command a flush command accompanied with an indication of said definitely transferred data words, the flush command flushing out from the modified FIFO the indicated number of data words and causes a repositioning of the modified FIFO's internal pointers so that the unflushed words are considered unread;
- step f) further includes a clear-and-flush command accompanied with a value stating the residual number of data definitely transferred to the external bus since the last issued flush command, the clear-and-flush command flushing out from the modified FIFO a number of data words equal to said residual number and causes a repositioning of the modified FIFO's internal pointers so that the unflushed words are considered unread.

42. An interface protocol in accordance with claim 40, wherein the assertion of terminate signal at the step e) is delayed to concede time at the peripheral FIFO for receiving

some other data from the macro-cell.

43. An interface protocol in accordance with claim from 37, wherein said remote filling status is saturated at an integer greater than or preferably equal to the maximum round-trip delay of a transaction through the common bus expressed in number of master clock cycles.

44. An interface protocol in accordance with claim 37, wherein if a peripheral monitoring code corresponding to one out of many possible correlated replies that a selected peripheral resource returns to the main module is indicating a peripheral resource of not prefetchable type, like a FIFO, and the calculated local filling status of the addressed peripheral resource is null, then a wait cycle opportunely long is introduced to allow in the meanwhile a possible increasing of residual room in the peripheral resource for transfer some other data in the current transaction, by effect of the activity of the connected macro-cell.

45. An interface protocol in accordance with claim 44, wherein when a selected resource is not prefetchable a step is executed for calculating a delta filling status corresponding to a detected positive variation into said returned remote filling status incurred between two consecutive steps; the delta filling status being added up to the previously defined local filling status.

46. An interface protocol in accordance with claim 37, wherein the protocol includes:

- a step indefinitely iterated for monitoring handshake signals and commands on the external bus and generating in correspondence a set of converted signals recognizable by the main module of the interface, and
- a step indefinitely iterated for converting into correspondent signals recognizable by an external protocol agent acting on the external bus an interface handshake signal and relevant protocol signals originated from the main module during said handshake steps for signalling that the main module is ready to transfer a datum or that a current transaction is terminated locally the interface.

47. An interface bus protocol in accordance with claim 46, wherein the protocol includes an indefinitely iterated general synchronization step in which: incoming address, the set of converted signals recognizable by the main module, the remote filling status, the filling status of the centralized buffer, other relevant signals events and states inside the interface and at the boundaries directly detectable on the main module and on its incoming and

outgoing common bus, are acquired and opportunely combined to match multiple logic conditions for issuing the right protocol commands and the relevant internal control signals relevant for executing said concurrent protocol steps for calculating delta and local filling status, executing read/write transactions towards the peripheral resources prefetchable and not prefetchable and handshaking them towards the external bus, and clear the centralized buffer.

48. An interface bus protocol in accordance with claim 37, wherein the protocol further includes the following steps:

- an internal arbitration step among transaction requests issued from transaction requesters embedded into the macro-cells and connected to peripheral resources of the interface that forward the requests to a centralized internal arbiter for executing a weighted Round Robin arbitration algorithm which selects, one transaction request at a time;
- as soon as one transaction request is selected a centralized request signal is asserted to enforce the main module to compete before an external arbiter for the master-ship of the external bus;
- an indefinite number of wait steps for monitoring on the external bus the arrival of handshake signals indicating the main module either master of the external bus or the master-ship assigned to an external bus protocol agent and the main module slave consequently.

49. An interface bus protocol in accordance with claim 37, wherein the protocol further includes the following concurrent steps:

- as soon as a grant signal indicating the main module master of the external bus is detected, the grant signal is prolonged to the peripheral resource, either prefetchable or not, associated to the selected transaction request;
- at the only assertion of the grant signal a grant command is activated to start a grant cycle spanning few master clock cycles to retrieve useful DMA parameters from the respective transaction requester located into the user macro-cell associated to the granted peripheral resource; as the DMA parameters are retrieved they are forwarded to the main module crossing the centralized buffer and stored into as many DMA registers, queued to the DMA parameters a read-and-query command is looped for filling up the centralized buffer;
- being the main module master of the external bus and the centralized buffer full, the read

transaction prosecutes and each looped read-and-query command while transfers to the external bus one datum extracted from the centralized buffer concurrently reads a datum from the granted peripheral resource and pushes it into the centralized buffer.

50. An interface bus protocol in accordance with claim 37, wherein the protocol further includes the following concurrent steps:

- as soon as one transaction request is selected from the internal arbiter an internal grant signal is simultaneously asserted and assigned to the peripheral resource whose transaction request has been selected by the granting algorithm;
- at the only assertion of the grant signal a grant command is activated to start a grant cycle spanning few master clock cycles to retrieve useful DMA parameters from the respective transaction requester located into the user macro-cell associated to the granted peripheral resource; as the DMA parameters are retrieved they are forwarded to the main module crossing the centralized buffer and stored into as many DMA registers, queued to the DMA parameters a read-and-query command is looped for filling up the centralized buffer;
- as soon as a grant signal is detected indicating the main module master of the external bus and being the centralized buffer full, the read transaction prosecutes and each looped read-and-query command while transfers to the external bus one datum extracted from the centralized buffer, concurrently reads a datum from the granted peripheral resource and pushes it into the centralized buffer;
- as soon as a grant signal is detected indicating an external bus protocol agent master of the external bus and being the centralized buffer full, the main module answer by asserting a RETRAY signal on the external bus to declare its temporary impossibility to accept transactions and iterating the RETRAY signal until a favorable grant is detected.

51. An interface bus protocol in accordance with claim 41, wherein the protocol further includes the following concurrent steps:

- as soon as one transaction request is selected from the internal arbiter an internal grant signal is simultaneously asserted and assigned to the modified FIFO whose transaction request has been selected by the granting algorithm;
- at the only assertion of the grant signal a grant command is activated to start a grant cycle spanning few master clock cycles to retrieve useful DMA parameters from the respective transaction requester located into the user macro-cell associated to the granted modified

FIFO; as the DMA parameters are retrieved they are forwarded to the main module crossing the centralized buffer and stored into as many DMA registers; queued to the DMA parameters a read-and-query command is looped for filling up the centralized buffer;

- as soon as a grant signal is detected indicating the main module master of the external bus and being the centralized buffer full, the read transaction prosecutes and each looped read-and-query command while transfers to the external bus one datum extracted from the centralized buffer, concurrently reads a datum from the granted modified FIFO and pushes it into the centralized buffer;
- as soon as a grant signal is detected indicating an external bus protocol agent master of the external bus and being the centralized buffer full, the main module issues a clear-and-flush command accompanied with a parameter-flag and with a null value of words transferred on the external bus, to cause a repositioning of the modified FIFO's internal pointers so that the data words transferred to the centralized buffer are considered unread and the DMA parameters are stored into the respective registers.

52. A software facility to write an operating system independent driver code suitable for macro-cell based devices interfaced to a microprocessor, or local bus; the driver code including a map file for assigning a different physical address to each distinguishable symbolic address included in a declaration part collecting arguments of a plurality of basic functions building up the complete driver code of the device macrocell-by-macrocell, wherein the map file is structured to match the architecture of the claimed microprocessor interface whose main module puts in communication the microprocessor to an internal bus of the interface to which a cluster of standardizable peripheral modules of the interface is appended, and each peripheral module being partitioned into subsets of homogeneous resources including a subset devoted to a stack of registers connected point-to-point with one or more interfaced macro-cells, and the absolute address assigned to a symbolic register address being calculated by means of the following expression:

$$\text{address} = \text{device\_base\_address} + (\text{leaf\_id} \times \text{Leaf\_add\_span}) + \text{Local\_register\_offset}$$

where: device_base_address indicates a reference point for addressing drivers of different devices; leaf_id indicates the starting address of an address span charged to a peripheral module; leaf_add_span is a constant span size; and local_register_offset is the displacement of the considered register from the starting address of the span.

53. A software facility in accordance with claim 52, wherein each single bit of an

addressed register having said absolute address is also symbolically addressable to be read or written.